APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/842,527	04/25/2001	Christopher L. Anderson	MS158543.1	7282

27195	7590	09/06/2005

AMIN & TUROCY, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114

EXAMINER
KANG, INSUN

ART UNIT	PAPER NUMBER
2193	

DATE MAILED: 09/06/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

-- *The MAILING DATE of this communication appears on the cover sheet with the correspondence address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE _3_ MONTH(S) FROM
THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed
  after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
  Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any
  earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on _21 June 2005_.

2a)☐ This action is **FINAL**.   2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is
   closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) _1-49_ is/are pending in the application.

   4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) _1-49_ is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

   Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

   Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   a)☐ All   b)☐ Some * c)☐ None of:

   1.☐ Certified copies of the priority documents have been received.

   2.☐ Certified copies of the priority documents have been received in Application No. _____.

   3.☐ Copies of the certified copies of the priority documents have been received in this National Stage
       application from the International Bureau (PCT Rule 17.2(a)).

   * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
      Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413)
      Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1.   This action is in response to the amendment filed 6/21/2005.

2.   As per applicant's request, claims 1, 9-11, 14, 19-23, 29-36, 39-43, and 47-49

have been amended. Claims 1-49 are pending in the application.

Claim Rejections - 35 USC § 101

3.   35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or
> composition of matter, or any new and useful improvement thereof, may obtain a patent therefor,
> subject to the conditions and requirements of this title.

4.   Claims 36-39, 43-46, and 48-49 are rejected under 35 U.S.C. 101 because the

claimed invention is directed to non-statutory subject matter.

Claims 36-39, 43-46, and non-statutory because they are directed to a

system without recitation of a computer or a computer-readable medium embodying the

claimed program elements. Although the independent claims start out reciting a

"system," the system does not have structural elements. The classes, compiler,

interface, code generator, etc that the system comprises are disembodied

arrangements so as to be called a "computer program" or compilation of facts,

information, or data *per se*, without creating any functional interrelationship, either as

part of the stored data or as part of the computing processes performed by the

computer ("acts") or computer readable medium so as to enable the computer to

perform the claimed elements such as converting the neutral representation into a high-

level language code as recited. With no other structure in the independent claims to

rely on, the alleged "system" of the independent claims turns out to be non-statutory for

being a computer program per se. Thus the claims represent non-functional descriptive

material that is not capable of producing a useful result, and hence represent only

abstract ideas. Therefore, the claims are non-statutory.

Claim Rejections - 35 USC § 103

5.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set
> forth in section 102 of this title, if the differences between the subject matter sought to be patented and
> the prior art are such that the subject matter as a whole would have been obvious at the time the
> invention was made to a person having ordinary skill in the art to which said subject matter pertains.
> Patentability shall not be negatived by the manner in which the invention was made.

6.   Claims 1-49 are rejected under 35 U.S.C. 103(a) as being unpatentable over

Bosworth et al. (US Patent 6,738,968) hereinafter referred to as "Bosworth" in view of

Dyer ("Java Decompiles compared," Java World, 7/1997).

Per claim 1:

Bosworth discloses:

- a language-neutral representation of a compile unit transformable to at least

one of a plurality of different types of code representations (Bosworth, i.e. "the common

language output files... have executable instructions in a "common"... in the sense of

universal... intermediate language suitable for representing the concepts of a plurality of

different types of source languages... so that only one type of intermediate language

need be used regardless of the specific source language used," col. 13 lines 27-47)

-a hierarchal arrangement of program elements that neutrally characterize

the compile unit (i.e. "the association of metadata with the boxed value type permits the

storage of the boxed value type within the object class hierarchy," col. 13 lines 1-10 and

abstract)

- at least one of the program elements representing a type declaration that

characterizes at least one class of programmatic constructs of the compile unit (i.e. "The

objects of a class hierarchy, such as class hierarchy...typically derive from a base root

object," col. 11 lines 27-44).

Bosworth does not explicitly teach converting the language neutral

representation of the compile unit to a corresponding representation of the compile unit

in at least one high-level language code. However, Dyer teaches that such decompiling

method was known in the art of software development, at the time applicant's invention

was made, to convert a compiled code into the original source code (page 1). It would

have been obvious for one having ordinary skill in the art of computer software

development to modify Bosworth's disclosed system to use a decompiler. The

modification would be obvious because one having ordinary skill in the art would be

motivated to "reverse-engineer compiled code (page 1, summary)" as suggested by

Dyer.

Per claim 2:

The rejection of claim 1 is incorporated, and further, Bosworth discloses:

-a collection of at least one member that characterizes programmatic attributes

associated with and able to be implemented within the at least one class (Bosworth, i.e.

"child boxed value types ...inherit methods and other attributes from parent boxed value

types...The dual representation of value types...as boxed value types in the object

class hierarchy implies that value types can have methods and can behave as objects,"

col. 12 lines 1-29) as claimed.

Per claim 3:

The rejection of claim 2 is incorporated, and further, Bosworth discloses an expression

class within the at least one class (Bosworth, i.e. "In the object class hierarchy..., built-in

value types...and user-defined value types...are stored as any other object within the

object class hierarchy..., providing the boxed value type with object-like attributes," col.

11 lines 45-62) as claimed.

Per claim 4:

The rejection of claim 2 is incorporated, and further, Bosworth discloses a statement

class within the at least one class (Bosworth, i.e.col. 7 lines 13-35) as claimed.

Per claim 5:

The rejection of claim 2 is incorporated, and further, Bosworth discloses a namespace

that contains the at least one class (Bosworth, i.e.col. 12 lines 60-67) as claimed.

Per claim 6:

The rejection of claim 1 is incorporated, and further, Bosworth discloses at least one of

the program elements of the hierarchal arrangement encapsulates another of the

program elements (Bosworth, i.e. col. 11 lines 27-44) as claimed.

Per claim 7:

The rejection of claim 1 is incorporated, and further, Bosworth discloses the interface

being operative to enable transformation of the language-neutral representation to a .

corresponding desired code representation (Bosworth, i.e. "The front end compiler...in

addition to being able to read and analyze their respective source files...are capable of

reading and analyzing files represented in the common language," col. 13 lines 27-47;

col 11. lines 27-44) as claimed.

Per claim 8:

The rejection of claim 7 is incorporated, and further, Bosworth discloses that the

program elements comprise objects, each object exposing at least one of a method,

attribute, and property of each respective object, the interface being operative to employ

the at least one of method, attribute and property to facilitate the transformation into the

desired code representation (i.e. "The front end compiler...in addition to being able to

read and analyze their respective source files...are capable of reading and analyzing

files represented in the common language," col. 13 lines 27-47; "metadata provides a

common interchange mechanism for use between tools that manipulate objects," col. 12

lines 39-59) as claimed.

Per claim 9:

The rejection of claim 7 is incorporated, and further, Bosworth discloses a compiler

interface programmed to enable transformation of the language-neutral representation

to a corresponding representation in a low-level language code (Bosworth, "any translation of the common language file into a form suitable for use by the runtime environment...convert the received common output files...into output code that can be executed in the execution environment," col. 14 lines 1-10) as claimed.

Per claim 10:

The rejection of claim 9 is incorporated, and further, Bosworth discloses an assembly of computer-executable instructions (Bosworth, i.e. "any translation of the common language file into a form suitable for use by the runtime environment...convert the received common output files...into output code that can be executed in the execution environment," col. 14 lines 1-10) as claimed.

Per claim 11:

The rejection of claim 7 is incorporated, and further, Bosworth discloses a code generator interface programmed to enable conversion of the language-neutral representation to a plurality of corresponding representations, wherein each representation is in a different high-level language code (Bosworth, i.e. col. 13 lines 27-47 ;"any translation of the common language file into a form suitable for use by the runtime environment...convert the received common output files...into output code that can be executed in the execution environment," col. 14 lines 1-10) as claimed.

Per claim 12:

The rejection of claim 1 is incorporated, and further, Bosworth discloses

-the program elements comprise instances of a plurality of language-neutral classes, ·

each instance defining an associated object (i.e. Fig. 3b and col. 11 lines 27-44) as

claimed.

Per claim 13:

The rejection of claim 12 is incorporated, and further, Bosworth discloses

at least one associated object represents the type declaration, at least another object

being encapsulated within the at least one object representing the at least one type

declaration, the at least another object representing program code of the compile unit

that derives from a class associated with the at least type declaration (i.e. Fig. 3b, "The

objects of a class hierarchy, such as class hierarchy..., typically derive from a base root

object," col. 11 lines 27-44) as claimed.

Per claim 14:

Bosworth discloses:

-A language-neutral representation of compile unit (Bosworth, i.e. "the common

language output files...have executable instructions in a "common"...in the sense of

universal...intermediate language suitable for representing the concepts of a plurality of

different types of source languages...so that only one type of intermediate language

need be used regardless of the specific source language used," col. 13 lines 27-47)

-an instance of at least one of a plurality of language-neutral classes, the plurality of

classes representing different programmatic constructs of a compile unit and having a

hierarchal relationship relative to each other, whereby transformation of the instance

into a different representation of the respective programmatic construct is facilitated

("The objects of a class hierarchy, such as class hierarchy...typically derive from a base

root object," col. 11 lines 27-44; "the association of metadata with the boxed value type

permits the storage of the boxed value type within the object class hierarchy," col. 13

lines 1-10; "any translation of the common language file into a form suitable for use by

the runtime environment...convert the received common output files...into output code

that can be executed in the execution environment," col. 14 lines 1-10).

Bosworth does not explicitly teach converting the instance of ...language neutral

... to a corresponding ...representation ... in at least one high-level language code.

However, Dyer teaches that such decompiling method was known in the art of software

development, at the time applicant's invention was made, to convert a compiled code

into the original source code (page 1). It would have been obvious for one having

ordinary skill in the art of computer software development to modify Bosworth's

disclosed system to use a decompiler. The modification would be obvious because one

having ordinary skill in the art would be motivated to "reverse-engineer compiled code

(page 1, summary)" as suggested by Dyer.

Per claim 15:

The rejection of claim 14 is incorporated, and further, Bosworth discloses

-each instance of a corresponding class of the plurality of classes represents a

respective programmatic construct of the compile unit, the plurality of instances being

organized in a hierarchal relationship based on the classes associated with the plurality

of instances and relationships among the programmatic constructs represented thereby

(Bosworth, i.e. col. 11 lines 27-44) as claimed.

Per claim 16:

The rejection of claim 15 is incorporated, and further, Bosworth discloses that each of

the plurality of instances exposes at least one item associated with the programmatic

construct represented thereby (Bosworth, i.e. col. 11 lines 27-44) as claimed.

Per claim 17:

The rejection of claim 16 is incorporated, and further, Bosworth discloses

- at least one of the plurality of instances represents a type declaration, at least another

instance being encapsulated within the instance representing the type declaration, the

at least another instance representing a programmatic construct that derives from the at

least type declaration(Fig. 3b, "The objects of a class hierarchy, such as class

hierarchy..., typically derive from a base root object," col. 11 lines 27-44).

Per claim 18:

The rejection of claim 17 is incorporated, and further, Bosworth discloses

at least one of a statement and an expression(Bosworth, "In the object class

hierarchy..., built-in value types... and user-defined value types... are stored as any

other object within the object class hierarchy..., providing the boxed value type with

object-like attributes," i.e. col. 11 lines 45-62; col. 7 lines 13-35) as claimed

Per claim 19:

The rejection of claim 16 is incorporated, and further, Bosworth discloses an interface

that enables transformation of the language neutral representation to the different

representation, the interface employs the at least one item to facilitate the

transformation of the language-neutral representation into the different representation

(Bosworth, i.e. col. 13 lines 27-47;"any translation of the common language file into a

form suitable for use by the runtime environment...convert the received common output

files...into output code that can be executed in the execution environment," col. 14 lines

1-10) as claimed.

Per claim 20:

The rejection of claim 19 is incorporated, and further, Bosworth discloses a compiler

interface programmed to enable transformation of the language-neutral representation

to the corresponding different representation in a low-level language code (Bosworth,

i.e. "any translation of the common language file into a form suitable for use by the

runtime environment...convert the received common output files...into output code that

can be executed in the execution environment," col. 14 lines 1-10) as claimed.

Per claim 21:

The rejection of claim 20 is incorporated, and further, Bosworth discloses an assembly

of computer-executable instructions (Bosworth, i.e. "any translation of the common

language file into a form suitable for use by the runtime environment...convert the

received common output files...into output code that can be executed in the execution

environment," col. 14 lines 1-10) as claimed.

Per claim 22:

The rejection of claim 19 is incorporated, and further, Bosworth discloses a code

generator interface programmed to generate a plurality of corresponding

representations from the language-neutral representation, wherein each representation

is in a different high-level language code from the language-neutral representation

(Bosworth, i.e. col. 13 lines 27-47 ;"any translation of the common language file into a

form suitable for use by the runtime environment... convert the received common output

files...into output code that can be executed in the execution environment," col. 14 lines

1-10) as claimed.

Per claim 23:

Bosworth discloses:

-A language-neutral representation of ...transformable to at least one other type of

software code representation (Bosworth, i.e. "the common language output files...have

executable instructions in a "common"... in the sense of universal...intermediate

language suitable for representing the concepts of a plurality of different types of source

languages...so that only one type of intermediate language need be used regardless of

the specific source language used," col. 13 lines 27-47)

-a hierarchal arrangement of objects, each object representing a different program

element of the compile unit class (Bosworth, i.e. "In the object class hierarchy..., built-in

value types...and user-defined value types...are stored as any other object within the

object class hierarchy..., providing the boxed value type with object-like attributes," col.

11 lines 45-62)

-at least one class object that represents at least one defined class of program

elements of the compile unit(i.e. Fig. 3b, "The objects of a class hierarchy, such as class

hierarchy..., typically derive from a base root object," col. 11 lines 27-44)

-at least one member object associated with the at least one class object that

represents computer-executable instructions operable on at least some program

elements in the at least one defined class(Bosworth, i.e. "In the object class

hierarchy..., built-in value types...and user-defined value types...are stored as any

other object within the object class hierarchy..., providing the boxed value type with

object-like attributes," col. 11 lines 45-62; col. 7 lines 13-35) .

Bosworth does not explicitly teach converting the language neutral

representation of computer executable instructions into a corresponding representation

in at least one high-level language code. However, Dyer teaches that such decompiling

method was known in the art of software development, at the time applicant's invention

was made, to convert a compiled code into the original source code (page 1). It would

have been obvious for one having ordinary skill in the art of computer software

development to modify Bosworth's disclosed system to use a decompiler. The

modification would be obvious because one having ordinary skill in the art would be

motivated to "reverse-engineer compiled code (page 1, summary)" as suggested by Dyer.

Per claim 24:

The rejection of claim 23 is incorporated, and further, Bosworth discloses a namespace object that represents a namespace of the compile unit, the namespace object comprising a collection of class objects including the at least one class object (Bosworth, i.e. col. 12 lines 60-67) as claimed.

Per claim 25:

The rejection of claim 24 is incorporated, and further, Bosworth discloses a plurality of member objects associated with the at least one class object, wherein the at least one class object represents a common base class that is shared by the plurality of member objects (i.e. "The objects of a class hierarchy, such as class hierarchy...typically derive from a base root object," col. 11 lines 27-44) as claimed.

Per claims 26-31, these claims are another versions of the claimed representation discussed in claims 18-22, respectively, and are rejected for the same reasons set forth in connection with the rejections of claims 18-22 above.

Per claims 32-35, these claims are another versions of the claimed representation discussed in claims 18-22, respectively, and are rejected for the same reasons set forth in connection with the rejections of claims 18-22 above.

Per claims 36-39, these claims are system versions of claims 23-26 and 31,

respectively, and are rejected for the same reasons set forth in connection with the

rejections of claims 23-26 above.

Per claim 40, it is the computer-readable medium version of claim 14,

respectively, and is rejected for the same reasons set forth in connection with the

rejection of claim 14 above.

Per claim 41, 42, it is the method version of claim 39, respectively, and is

rejected for the same reasons set forth in connection with the rejection of claim 39

above.

Per claims 43-45 and 46, these claims are system versions of claims 37 and 38

respectively, and are rejected for the same reasons set forth in connection with the

rejections of claims 37 and 38 above.

Per claim 47, it is the method version of claim 29, respectively, and is rejected for

the same reasons set forth in connection with the rejection of claim 29 above.

Per claim 48 and 49, it is the system version of claim 30, respectively, and is

rejected for the same reasons set forth in connection with the rejection of claim 30

above.

Response to Arguments

7.   Applicant's arguments filed 6/21/2005 have been fully considered but they are

not persuasive.

Per claims 1, 14, 23, 32, 36, 40, 41, 43, 47, and 48:

The applicant admits that Bosworth teaches "an intermediate representation suitable for

representing the concepts of a plurality of different source languages" and the "cited

prior art further teaches conversion of the intermediate representation into a low level

language code(remark, 13)." However, the applicant states, "Bosworth, et al. is silent

regarding converting the language neutral representation into a high level language

code."

In response, Applicant's arguments with respect to claims above have been

considered but are moot in view of the new ground(s) of rejection.

8.   Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Insun Kang whose telephone number is 571-272-3724.

The examiner can normally be reached on M-F 9:30-6.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Kakali Chaki can be reached on 571-272-3719. The fax phone number for

the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free).

Any inquiry of a general nature or relating to the status of this application should

be directed to the TC 2100 Group receptionist: 571-272-2100.

I. Kang
AU2193

ANIL KHATRI
PRIMARY EXAMINER